Leveraging the Ubiquitous Web as a Secure Context-aware Platform for Adaptive Applications

Heiko Desruelle, Frank Gielen
Dept. of Information Technology – IBCN
Ghent University – IBBT
Ghent, Belgium
{*heiko.desruelle, frank.gielen*}*@intec.ugent.be*

John Lyle
Dept. of Computer Science
University of Oxford
Oxford, UK
john.lyle@cs.ox.ac.uk

***Abstract*—The availability as well as diversity of connected devices has turned the Internet into a ubiquitous concept. In addition to desktop and laptop PCs, the Internet also connects numerous mobile devices, home entertainment systems, and even in-car units. Various new types software applications arise, trying to make optimal use of this trend. However, as the fragmentation of devices and platforms grows, application developers are increasingly facing the need to cover a wider variety of target devices. Maintaining a viable balance between development costs and market coverage has turned out to be a challenging issue when developing applications for such a ubiquitous ecosystem. In this paper, we present the Webinos approach, a distributed web runtime that adaptively leverages the device-independent characteristics of the Web. By introducing the concept of context-aware Personal Zones, the Webinos platform aims to facilitate the development of self-adaptive and immersive applications, optimized for ubiquitous computing environments.**

***Keywords*-ubiquitous web; context-aware platform; distributed runtime; adaptive applications; webinos**

I. Introduction

The Internet is drastically changing the way people work and live. As the diversity of connected devices is increasing rapidly, the Internet is penetrating our everyday lives through a multitude of devices. From desktop and laptop PCs, to mobile devices, to home entertainment systems and even in-car headunits, users throughout all consumer segments should prepare for a connected experience [1]. The evolution towards a ubiquitous Internet creates the opportunity for numerous new and innovative software applications. The main driver for such applications would be to seamlessly enable the inherently nomadic character of a ubiquitous system. Furthermore, this driver should aim to enable users to access and share information whenever and wherever they want, regardless of the device type that is being used to initiate the operation.

The development and deployment of applications for such a ubiquitous ecosystem, however, introduces an important series of resource-consuming requirements [2]. The available combinations of hardware characteristics, operating systems, software frameworks, etc. are virtually endless. For software developers, this diversity has turned out to be a double-barreled asset. It provides consumers the freedom to operate applications at will across several devices. On the other hand, the device diversity asset heavily fragments the application's delivery targets. By the absence of a general native development solution, developers often have no alternative than to create and maintain a set of device-dependent versions of their applications. Hence, ensuring a viable balance between development costs and an application's market coverage will more than ever become a challenging issue.

Against this backdrop, the use of web technologies for application development purposes has proven to be a viable long-term candidate solution [3]. Through years of standardization efforts and the wide adoption of languages such as HTML, CSS, and JavaScript, the web can be deployed as a powerful foundation for universal application development and delivery. Running on top of the Internet infrastructure, the web application ideology is rapidly gaining momentum amongst developers.

A web-based application development approach has been explored from various perspectives. Developers can opt for pure web applications, running in a standard browser environment. However, due to the sandboxed nature of browsers this approach drastically limits the available APIs (Application Programming Interfaces) to the underlying device. In turn, a hybrid web application approach was introduced providing developers access to a richer API set, whilst still maintaining most of the cross-platform advantages from pure web applications. This type of application is still built using web technology, but no longer uses the browser as the client-side runtime environment. A separate client-side web runtime framework is deployed to bridge the gap between native and web applications by granting the application scripting access to most device APIs. Hybrid web applications are currently being developed using web widget engines such as provided by the BONDI/WAC [4] initiatives, device-independent frameworks such as the PhoneGap [5] application wrapper, and even completely web-centric operating systems such as Chromium OS [6] and HP webOS [7].

Current hybrid web application solutions, however, only

partially succeed in enabling a convincing ubiquitous experience [8]. Their main focus lies with porting traditional API support and operating system aspects to the web. Applications built upon these old principles result in virtual silos, unable to truly cross the physical boundaries of a device. By neglecting the evolution towards, e.g., distributed user interfaces, adaptive context-aware application behavior, etc. the true immersive nature of ubiquitous computing is mostly left behind [9] [10]. The absence of elaborate context-awareness is a key element driving this issue. In order for ubiquitous applications to adaptively support various contextual situations, the underlying application platform needs to provide structured, as well as secure and up-to-date access to the user's contextual setting. This requirement is not just limited to providing access to a detailed description of the target delivery context (screen size, interaction methods, available sensors, etc). Moreover, structured access to details regarding the user context (personal preferences, social context, disabilities, etc.) and the physical environment (location, time, etc.) ought to be supported.

From this perspective, we introduce the Webinos approach, a platform aiming to support hybrid web applications across mobile, PC, home media and in-car devices. Structured along a federated hierarchy, the proposed architecture enables developers to access a common set of rich context-aware APIs, allowing applications to dynamically adapt their cross-user, cross-service, and cross-device functionality in an open yet secure manner.

The remainder of this paper is structured as follows. Section II discusses background and related work. Section III provides a general overview of the proposed federated application platform. Section IV elaborates on the platform details of setting up secure context-awareness support. Section V discusses the use case of an adaptive social networking application. Finally, the conclusion and future work is outlined in Section VI.

II. Background and Related Work

The availability of detailed and reliable metadata regarding a user's contextual situation provides an important driver for enabling rich ubiquitous applications. The exact entities represented by this contextual information can be of a very dynamic nature, potentially affecting the consumer's expectations towards the application's user interface, behavior, content, etc. In initial context-aware research, context of use was considered a component containing only two parameters: the end-user's location and the set of objects in the immediate vicinity [11]. The subsequent introduction of extensible contextual categories has drastically increased the flexibility of this definition. Chen and Kotz hereto identified five contextual base categories: the device context, the user context, the environment context, the time context, and the historical context [12].

The device context describes the characteristics of the target device that is being used to access the application. A ubiquitous ecosystem covers a diversity of screen sizes, interaction methods, software support, etc. In web-based environments, the device capabilities are generally retrieved through Resource Description Framework (RDF) devices profiles, i.e., User Agent Profile (UAProf) [13] and Composite Capability/Preference Profiles (CC/PP) [14]. The necessary device identification step in this process is handled through HTTP header user agent matching. In order to facilitate the collection and aggregation of these device profiles, the W3C Mobile Web Initiative (MWI) standardized the Device Description Repository specification (DDR). The specification provides an API and its associated vocabulary for structured access to context providers services [15]. In essence, a DDR thus provides a standardized means for retrieving contextual information about a-priori knowledge on the characteristics of a particular target device or web runtime. Various open as well as proprietary DDR implementations are actively being maintained. Most notably OpenDDR, WURFL, and DeviceAtlas.

In a ubiquitous setting, the end-user's profile description gains more and more importance. Besides exposing information on user preferences and specific experience, this model should also comprise knowledge regarding the user's specific abilities and disabilities, e.g., enabling accessibility requirements for providing support to elderly people, and people with disabilities. From this perspective, Heckmann proposed the GUMO formalism as a general user model ontology for representing generic user descriptions using the Web Ontology Language semantics (OWL) [16]. The current challenge in this domain is modeling the enormous amount of parameters and relationships that characterize the user context [17]. To overcome this issue, forces are being joined with other ontology-driven projects such as Linked Data [18], and UbisWorld [19].

The environment-, time-, and historical context aspects define where, how, and when the interaction between the user and an application is exactly taking place. The environment context is specified by observing the numerous sensors available on the user's device (e.g., location, temperatures, network service discovery, the level of background noise, etc.). Furthermore, the notion of time and historical context is not to be neglected. As context is a dynamic concept, support for temporal patterns recognition and management is needed. The W3C Ubiquitous Web Domain is currently in the process of standardizing the Delivery Context Ontology specification (DCO) [20]. The DCO provides a formal model of the characteristics of the environment in which devices, applications, and services are operating.

III. Webinos Hybrid Application Platform

In order to enable application developers to set up services that fade out the physical boundaries of a device, we

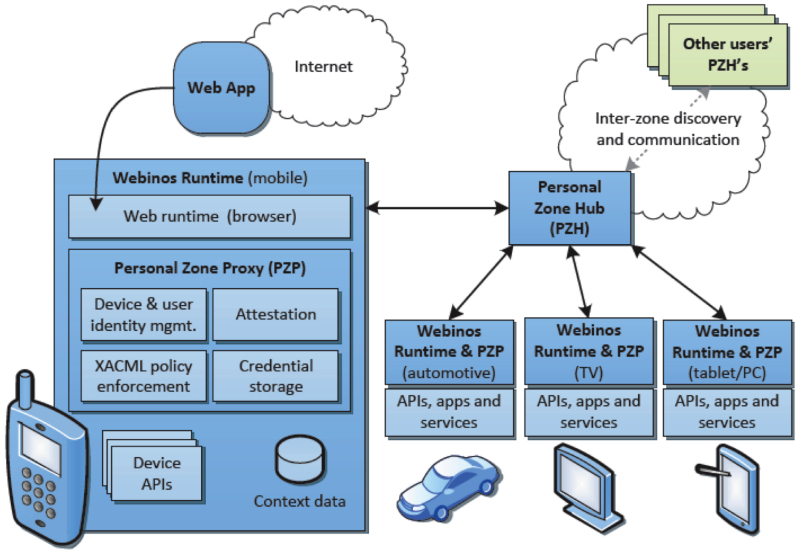

Figure 1. High-level Webinos platform overview

propose the Webinos architecture. Webinos is a federated web application platform and its runtime components are distributed over the devices, as well as the cloud. Figure 1 depicts a high-level overview of the platform's structure and deployment. The system's seamless interconnection principle is cornered around the notion of a so called Personal Zone. The Personal Zone represents a secure overlay network, virtually grouping a user's personal devices and services. To enable external access to and from the devices and services in this zone, the Webinos platform defines centralized Personal Zone Hub (PZH) components. Each user has his own PZH instance running in the cloud. The PZH is a key element in this architecture, as it contains a centralized repository of all contextual data in the Personal Zone. Moreover, the PZH keeps track of all devices and services in the zone and provides functionality to enable their mutual communication. This way, the PZH facilitates cross-device interaction with someone's services over the Internet. The PZHs are federated, allowing applications to easily discover and share data and services.

On the device-side, a Personal Zone Proxy (PZP) component is deployed. The PZP handles the direct communication with the zone's PZH. In order to keep the user's Personal Zone synchronized, the PZP is responsible for communicating device status with its PZP. This communication channel is built around a publisher-subscriber pattern [21]. As all external communication goes through the PZP, this component also acts as a policy enforcement point by managing all access to the device's exposed resources. In addition, the PZP is a fundamental component in upholding the Webinos platform's offline usage support. Although the proposed platform is designed with a strong focus on taking benefit from online usage, all devices in the Personal Zone have access to a locally synchronized copy of the data maintained by the PZH. The PZP can thus act in place of the PZH in case no reliable Internet connection can be established. This allows users to still operate the basic functionality of their applications even while being offline. All data to and from the PZP is again synchronized with the PZH as soon as the Internet access gets restored.

The Web Runtime (WRT) represents the last main component in the Webinos architecture. The WRT can be considered as the extension of a traditional web browser engine (e.g., WebKit, Mozilla Gecko). The WRT contains all necessary components for running and rendering web applications specified using standardized web technologies: HTML parser, JavaScript engine, CSS processor, rendering engine, etc. Furthermore, the WRT maintains a tight binding with the local PZP. The WRT-PZP binding allows the WRT to be much more powerful than traditional browser-based application environments. Through this binding, applications running in the WRT are able to securely interface with local device APIs and services. In addition, the PZP also allows the runtime to connect and synchronize with other devices in the Personal Zone through its binding with the PZH.

IV. Secure Context-aware Personal Zone

The innovative nature of the proposed approach lies with the platform's capability to establish a cross-device, cross-service, cross-user overlay network. For this Personal Zone concept to be successfully adopted by ubiquitous application developers, the platform needs to provide these developer access to a rich at-runtime overview of the user's contextual setting. As stipulated in Section I, elaborate platform support for transparent context management is vital. In this section, we provide more detail on the available developer tools for

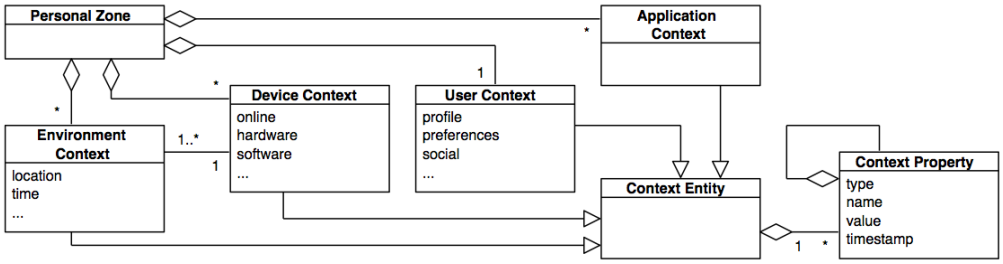

Figure 2. Simplified Webinos Personal Zone context model

setting up secure context-awareness within a Personal Zone environment.

A. Delivery Context Model

The Webinos delivery context model is defined to span all the platform's contextual knowledge within the user's Personal Zone. The model builds upon the W3C's Delivery Context Ontology (DCO) specification [20]. The Webinos delivery context model comprises four top-level submodels: the user context, the device context, the environment context, and the application context (see Figure 2 for a high-level overview). The first three submodels are internally managed and updated by the Webinos platform, whilst the application context model is to be maintained by the application developer. In order for each of these proposed models to support historical evaluation, pattern detection, conflict resolution strategies, all stored context properties are timestamped. The contextual information regarding the Personal Zone's owner is described by the user context model. This model consists of an aggregation of user profile data, user preferences, social context information, etc. Furthermore, each device and its physical environment are described by a separate instance of respectively the device context model and the environment context model. A device context model comprises knowledge regarding the corresponding device's availability in the Personal Zone, hardware characteristics, supported software, etc. The environment context model contains a description of a certain device's location, surrounding noise levels, etc. Lastly, the application context model provides developers the freedom to store a number of contextual properties, describing a situation from the perspective of their application.

B. Context Framework and API

The Webinos context framework is built on top of the above described context models. As depicted in Figure 1, providing application developers access to an elaborate distributed context framework is one of the core Webinos service. The Webinos context framework provides all necessary functionality for acquiring, storing, inferring new knowledge, and granting external access to the available contextualized data. Web applications running in the Webinos WRT, as well as other Webinos services, can rely on this framework to support their at-runtime need for contextualized data.

The Webinos Personal Zone is structured as a distributed system. In order to keep the zone synchronized, strong communication facilities between the device PZPs and the centralized PZH have architecturally been put in place. The Webinos context framework tries to makes optimal use of this structured communication channel to gain additional contextual knowledge regarding the Personal Zone. The context framework hooks into the PZP's event dispatching and synchronization mechanism. As visualized in Figure 3, out-bound status events are intercepted by the framework's context acquisition component and subsequently filtered for relevant data. The extracted context is locally stored and synchronized with the rest of the zone through a context-update event over the communication channel. The context acquisition process is autonomously managed by the Webinos platform and operates completely transparent for both the user and application developers. Moreover, the context framework is closely coupled with the PZP's security and policy enforcement framework. This binding ensures the secure handling of all context data that is being stored and accessed, as it often contains highly sensitive information.

For application developers aiming to create context-aware ubiquitous applications, the context framework provides an API to access Personal Zone wide context information. The context API supports the W3C standardized SPARQL RDF query language for unambiguously stating powerful context queries [22]. All context API requests are passed to the query processor component. The processor parses the request and checks its execution rights in collaboration with the PZP's policy enforcement framework. In case the request is granted by the PZP, the query is optimized and dispatched for execution. The API supports two modes for accessing context information: a generic query mode, and a change subscription mode. The generic query mode allows applications to execute targeted queries for specific context data in the storage system. The change subscription mode, on the other hand, enables an application to subscribe for specific context update events. These events are triggered by the context framework when new contextual knowledge is acquired.

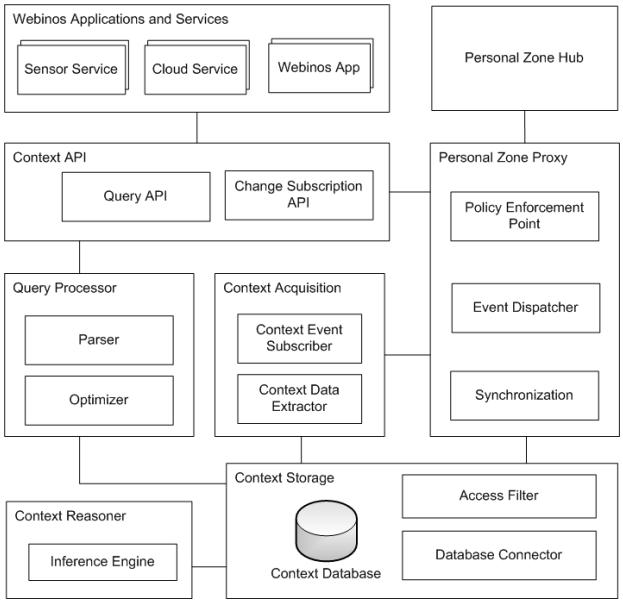

Figure 3. Webinos platform's distributed context framework, enabled through its tight integration with the Personal Zone

C. Policy Enforcement Support

The Webinos platform aims to meet the security and privacy requirements of applications and end users primarily through an access control policy system. Every access to a Webinos API is mediated by policies, enforced by the Personal Zone Proxies on each device as well as in the Personal Zone Hub. This action follows the principle of least privilege, granting applications only the permissions they require. Access policies are set when an application is first installed and can be updated subsequently. The policy system is derived from the BONDI/WAC architecture [4] and uses XACML (eXtensible Access Control Markup Language) including a number of extensions developed by the PrimeLife project [23]. XACML is a general-purpose access control language for defining policies based on subjects, resources, action and conditions [24]. By including the PrimeLife XACML extensions, the Webinos policy enhancement framework can allow users to specify detailed situation-specific access control policies. This is a significant advantage over current web runtime solutions and native mobile application platforms, where once an application has been granted access to a particular asset this access can be reused without further control.

Context data is often privacy-sensitive, as its analysis might reveal a user's history of actions or the people and devices they have interacted with. The Webinos platform aims to follow the principle of least surprise, so that a minimum of unexpected data disclosures will occur. This is achieved by disabling the collection of most context data by default, and providing the user a simple interfaces to turn it on again, complete with feedback about the kind of data that is being shared and stored. Where possible, data is filtered to remove unnecessary personal data. The main advantage of the Webinos platform is that context data remains within the zone and under the control of the end-user. This compares favorably to online user tracking, as users are able to view and manage the data stored about them, and applications will have to request specific access to this information.

V. Use Case

To elaborate on the application possibilities of the Webinos approach, we present a use case that has been built with the platform. The application is a cross-device social media app, able to use the APIs of television sets, mobile devices and desktop computers within a Personal Zone. The application utilizes the platform's default knowledge of a user's devices as well as their exposed capabilities and services. A user has the possibility to set policies for dispatching system API calls of the application to alternative devices. In result, the input (i.e., multimedia access, text input modality, contacts retrieval, etc.) and output (i.e., display selection) operations are adaptively abstracted from the traditional physical device level to the Personal Zone level. E.g., if a user wants to post a new message to one of his contacts on the Twitter social network: he can use his television set for displaying the main UI, use his smartphone an interaction device for navigating through the interface, putting in text, and accessing the device's contact list, access this home media center to attach a video to his post, etc.

A prototype of the proposed platform and use case are implemented and made available as part of the Webinos open source project [25]. Based on the project's extensive analysis of the current ubiquitous ecosystem [26], the following prototype platforms have been selected: PC (Linux, Windows, MacOS), mobile (Android), vehicles (Linux), home entertainment (Linux).

VI. Conclusion and Future Work

In this paper we presented the Webinos application platform approach, aiming to enable immersive ubiquitous software applications by leveraging the cross-platform possibilities of the web. The proposed approach utilizes the web infrastructure to establish its Personal Zone concept, a virtual overlay network for grouping all of user's devices and available services. Through the federated structure of Personal Zones, Webinos is able to provide application developers access to elaborate at-runtime context data regarding the current user, his devices, and the surrounding environment. The availability of this information allows developers to more accurately anticipate to a user's contextual situation. The Webinos platform's context-awareness enables numerous applications that make full use of the diversity and interconnectivity of devices. From this perspective, Webinos aims to be a key enabler in the realization of ubiquitous

applications that are able to execute across the physical boundaries of devices.

While the extensive evaluation of our approach has yet to be carried out, initial testing of prototype implementations shows promising results. Although the proposed platform addresses challenging issues in the ubiquitous application development domain, the current architecture only represents a first milestone in the pursuit of true ubiquitous application convergence. Whilst the Webinos platform provides structured access to rich contextual knowledge, it is still the application developers' responsibility to incorporate the necessary logic that allows their applications to act accordingly. Therefore, future work should include research on further extending the platform with (semi-) automated application adaptation mechanisms, driven by the platform's rich context-awareness. Regarding the privacy and security impact of such an application runtime, there will undoubtedly be a need to further experiment with user interfaces. This in order to strike an acceptable balance between the advantages that context sensitivity can offer, as well as privacy and user and developer convenience.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7-ICT-2009-5, Objective 1.2) under grant agreement number 257103 (Webinos project).

REFERENCES

[1] M. Tuters and K. Varnelis, "Beyond locative Media: giving Shape to the Internet of Things," *Leonardo*, vol. 39, no. 4, 2006, pp. 357-363.

[2] G. Banavar and A. Bernstein, "Challenges in design and software infrastructure for ubiquitous computing applications," *Advances in computers*, vol. 62, 2004, pp. 179-202.

[3] G. Lawton, "Moving the OS to the Web," *Computer*, vol. 41, no. 3, 2008, pp. 16-19.

[4] *Widget runtime high level technical specifications*, Wholesale Application Community (WAC) specification version 1, 2010.

[5] A. Charland and B. Leroux, "Mobile application development: web vs. native," *Communications of the ACM*, vol. 54, no. , 2011, pp. 49-53.

[6] W.T. Tsai, Q. Shao, X. Sun, and J. Elston, "Real-Time Service-Oriented Cloud Computing," in *Proc. IEEE 6th World Congr. on Services*, 2010, pp. 473-478.

[7] A. Weiss, "WebOS: say goodbye to desktop applications," *Networker*, vol. 9, no. 4, 2005, pp. 18-26.

[8] A. Taivalsaari and T. Mikkonen, "The Web as an Application Platform: The Saga Continues," in *Proc. IEEE 37th Conf. on Software Engineering and Advanced Applications*, 2011, pp. 170-174.

[9] H. Desruelle, D. Blomme, and F. Gielen, "Adaptive user interface support for ubiquitous computing environments," in *Proc. 2nd Int. Workshop on User Interface eXtensible Markup Language*, 2011, pp. 107-113.

[10] H. Desruelle, D. Blomme, and F. Gielen, "Adaptive mobile web applications through fine-grained progressive enhancement," in *Proc. 3rd Int. Conf. on Adaptive and Self-Adaptive Systems and Applications (ADAPTIVE)*, 2011, pp. 51-56.

[11] B. Schilit, N. Adams, and R. Want, "Context-aware Computing Applications," in *Proc. IEEE 1st Int. Workshop on Mobile Computing Systems and Applications*, 1994, pp. 85-90.

[12] G. Chen and D. Kotz, "A Survey of Context-aware Mobile Computing Research," Dept. Computer Science, Dartmouth College, Tech. Rep. TR2000-381, 2000.

[13] *WAG UAProf*, Wireless Application Protocol specification WAP-248-UAPROF-2001020, 2001.

[14] *Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0*, W3C recommendation, 2004.

[15] *Device Description Repository Core Vocabulary*, W3C working group note, 2008.

[16] D. Heckmann, "Ubiquitous User Modeling," Ph.D. dissertation, Dept. of Computer Science, Saarland University, 2005.

[17] J.L.T. Silva, A. Moreto Ribeiro, E. Boff, T. Primo, and R.M. Viccari, "A Reference Ontology for Profile Representation in Communities of Practice," *Metadata and Semantic Research*, vol. 240, 2011, pp. 68-79.

[18] T. Heath and C. Bizer, "Linked data: Evolving the web into a global data space," in *Synthesis Lectures on the Semantic Web: Theory and Technology*, vol. 1, no. 1. Morgan & Claypool, 2011, pp. 1-136.

[19] D. Heckmann, M. Loskyll, R. Math, P. Recktenwald, and C. Stahl, "Ubisworld 3.0: A Semantic Tool Set for Ubiquitous User Modeling," in *Proc. 17th Int. Conf. on User Modeling, Adaptation and Personalization*, 2009.

[20] *Delivery Context Ontology*, W3C working group note, 2010.

[21] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-oriented software architecture: A system of patterns*. West Sussex: John Wiley & Sons, 2001.

[22] T. Segaran, C. Evans, and J. Taylor, *Programming the Semantic Web*. Sebastropol: O'Reilly Media, 2009.

[23] C.A. Ardagna, S. De Capitani Di Vimercati, E. Pedrini, and P. Samarati, "Primelife policy language," in *Proc. W3C workshop on access control application scenarios*, 2009.

[24] *eXtensible Access Control Markup Language (XACML) version 1.1*, OASIS standard, 2003.

[25] Webinos, "Webinos Developer Portal," Available: https://developer.webinos.org/ [May. 1, 2012].

[26] Webinos, "Industry landscape, governance, licensing and IPR frameworks," Tech. Rep. D02.3, 2011.